PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 4 : G06F 13/36

A1

(11) International Publication Number: WO 88/ 08162

(43) International Publication Date: 20 October 1988 (20.10.88)

(21) International Application Number: PCT/US88/01236

(22) International Filing Date: 8 April 1988 (08.04.88)

(31) Priority Application Number: 036,613

(32) Priority Date: 10 April 1987 (10.04.87)

(33) Priority Country: US

(71) Applicant: EIP MICROWAVE, INC. [US/US]; 2731 North First Street, San Jose, CA 95134 (US).

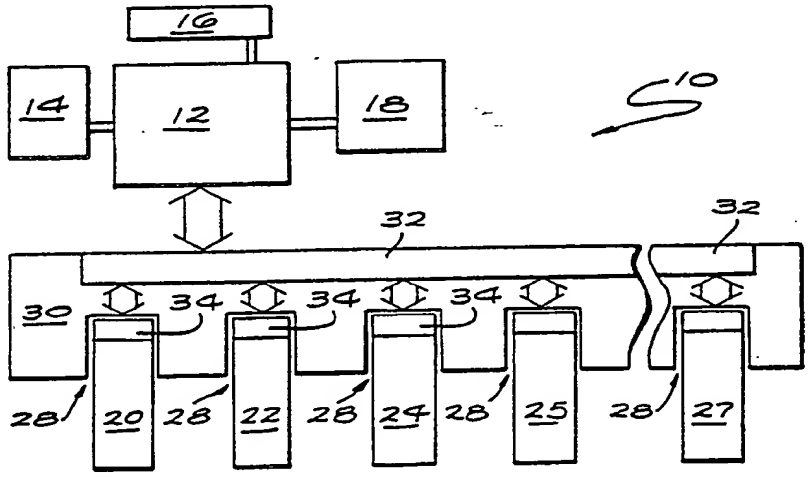
(72) Inventor: BARKHORDARIAN, Shahram, P. ; 4456 Hilton Avenue, San Jose, CA 95130 (US).

(74) Agents: ROBBINS, Billy, A. et al. ; Nilsson, Robbins, Dalgarn, Berliner, Carson & Wurst, 201 North Figueroa Street, Fifth Floor, Los Angeles, CA 90012 (US).

(81) Designated States: DE (European patent), FR (European patent), GB (European patent), IT (European patent).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: DATA TRANSFER SYSTEM FOR A MULTIPROCESSOR COMPUTING SYSTEM



(57) Abstract

A multiprocessor computing system including a plurality of output boards with one or more microcomputers (20, 22, 24) all contained upon printed circuit boards appropriately interconnected through a system bus (32). A host computer (12) utilizing a data flow logic operating system program through interaction with the user applies appropriate, sometimes customized, applications programs to define connection between the various printed circuit boards to accomplish data flow therebetween in accordance with the applications programs. The timing sequence for the various computational routines or tasks is automatically controlled by the length of the data pack as it is transferred in accordance with the applications programs so that the various tasks may be performed either serially or in parallel or a combination thereof as may be required for efficient communication between the PC boards as well as the specific portions contained thereon which perform the computational tasks.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	**FR**	France	**ML**	Mali
AU	Australia	**GA**	Gabon	**MR**	Mauritania
BB	Barbados	**GB**	United Kingdom	**MW**	Malawi
BE	Belgium	**HU**	Hungary	**NL**	Netherlands
BG	Bulgaria	**IT**	Italy	**NO**	Norway
BJ	Benin	**JP**	Japan	**RO**	Romania
BR	Brazil	**KP**	Democratic People's Republic of Korea	**SD**	Sudan
CF	Central African Republic			**SE**	Sweden
CG	Congo	**KR**	Republic of Korea	**SN**	Senegal
CH	Switzerland	**LI**	Liechtenstein	**SU**	Soviet Union
CM	Cameroon	**LK**	Sri Lanka	**TD**	Chad
DE	Germany, Federal Republic of	**LU**	Luxembourg	**TG**	Togo
DK	Denmark	**MC**	Monaco	**US**	United States of America
FI	Finland	**MG**	Madagascar		

DATA TRANSFER SYSTEM FOR A MULTIPROCESSOR COMPUTING SYSTEM

BACKGROUND OF THE INVENTION

The present invention relates to computer systems and more particularly to computer systems for processing data through the use of multiple processors.

Computer systems generally operate with one processor which handles all computional operations in "serial" fashion. However, since there are practical limits to the speeds at which processors may operate, these systems are themselves inherently limited in their capabilities. The alternative is to utilize multiple processors which process data in "parallel" fashion. In practice this has turned out to be difficult. However, two approaches have achieved some success in the past. Systems have been developed in which separate processors share the same memory. The effectiveness of a shared memory system is limited by the "bandwidth" of the memory. As the number of processors increases the processing time associated with coordinating access to the shared memory begins to consume a large portion of the available system time. Alternatively, systems have

been developed in which microprocessors are hardwired together in networks with each processor performing separate and distinct operations and communicating with certain other processors on separate bus lines. This system requires a complex communications structure and lacks flexibility to accomplish any processing job except the job for which it was specifically constructed.

The present invention helps to overcome the above-described limitations by providing a system which is not limited by memory bandwidth and is flexible in allowing the user to adjust the processing structure to the output desired and equipment available. Further, the present invention provides high processing speeds without the use of special processors or the complexity of a hardwired multiprocessor system.

SUMMARY OF THE INVENTION

The present invention constitutes a system for effectuating interprocessor communications in a multiprocessor computer system. The present invention includes a plurality of data processing units, a common system bus interconnecting the units and a bus arbitration network for controlling access to the system bus. The processor units each comprise a conventional microprocessor, RAM memory and a local bus. However, the processor units also include a buffer device for enabling a data communications pathway between the system bus and the local bus and data transmission logic which functions to implement the actual data transfers between processor unit. The processor units further include in memory application program modules for data processing and an

operating system program for providing initialization signals to the data transmission logic for initiating and controlling data transfers.

In operation, the applications modules on each processor unit contain system call instructions which direct the microprocessor to execute operating system program routines for the input and output of data between applications modules. If the applications modules are on different processor units the operating system generates initialization signals to initiate and control the data transmission logic. In response, the data transmission logic interacts with the bus arbitration network to secure access to the system bus, establishes contact with the destination processor unit, activates the buffer device, supplies data memory address signals and provides control and timing signals to govern the actual transfer. Concurrently, the data transmission logic on the destination processor acknowledges contact, actuates its buffer device and provides certain data memory address, control and timing signals to govern the transfer for its processor unit.

BRIEF DESCRIPTION OF THE DRAWINGS

FIGURE 1 is a block diagram of the overall computing system.

FIGUREs 2a and 2b are flowchart diagrams of an exemplary application program for a data processing job.

FIGURE 3 is a flowchart diagram of an operating system routine for sending data.

FIGURE 4 is a flowchart diagram of an operating system routine for receiving data.

FIGURE 5 is a simplified functional block diagram of the data transfer structure for a processor.

FIGURE 6 is a block diagram of the data transfer structure for sending data.

FIGURE 7 is a block diagram of the data transfer structure for receiving data.

DESCRIPTION OF THE PREFERRED EMBODIMENT

Referring now to FIGURE 1, a block diagram of the hardware for the overall computing system is illustrated. The computing system 10 includes a host computer 12 having RAM memory and a set of standard input and output ports including at least one parallel port. An IBM PC AT with 512K RAM memory would comprise a satisfactory computer. The computer 12 includes a floppy disc storage memory device 14 of conventional design for permanent storage of programs. The computer 12 also includes a keyboard 16 to allow input from the user and a video monitor 18 to allow output from the computing system 10 to be visually displayed.

The computing system 10 further includes a set of data pocessing units 20, 22, 24 and one or more input/output units such as I/0 boards 25 and 27. The units 20, 22 and 24 are microcomputers which each comprise a Motorola 68000 microprocessor or a related family member with 256K RAM memory mounted on a single processor board. The I/0 boards 25 and 27 may typically be analog to digital converters, synthesizer boards or some

other board for fulfilling a customized function. The boards all fit into connection slots 28 in a housing structure 30. The housing structure 30 includes a frame constructed to support fifteen connection slots suitable to connect up to fifteen separate processor boards into the computer system 10. The host structure 30 also includes a modified VME bus 32 which interconnects the data processing units 20, 22, 24 and I/O boards 25 and 27 and connects the host computer 12 to the data processing units 20, 22, 24. The VME bus is modified to include a number of lines for the control of data transfers between processors and I/O boards. The data processing computers 20, 22 and 24 and the system computer 12 are interfaced to the VME bus by a digital logic system for coordinating and controlling interprocessor communications over the bus 32 having components 34 associated with each processor unit.

The data processing units 20, 22 and 24 are all controlled by a DFL ("Data Flow Logic") operating system program which is downloaded from the host computer 12. Initially, the host computer 12 contains a DFL compiler program and a DFL operating system program. The DFL compiler program allows the user to compose applications programs and define connectivity between these programs prior to their being loaded into the data processing units 20, 22 and 24. The DFL operating system program provides a mechanism for actuating communications between the units 20, 22 and 24 and for implementing data flow into and out from the units 20, 22 and 24 in accordance with the applications programs and their connectivity as supplied by the user.

Most data processing jobs involve a number of computional routines of varying complexity which may be expressed in computer code as program modules and which may be linked together to form a program for accomplishing the processing job. Referring now to FIGURE 2a, a flowchart is shown for such a program with nine serially connected routines, such as compute X or format Y, in the form of a program structured for execution by a digital computer having one processor. Ordinarily, the routines would all run on the computer in sequential fashion.

However, the computional routines which make up a processing job may be linked together differently. Frequently, routines are not required to be serially connected and may in fact be performed in parallel since the routines are not dependent on each other for the supply of required input. In accordance with the above, it is useful to view computational routines and their program modules as "tasks" and the links between these tasks as "arcs" which represent channels for data flow between the tasks. The tasks and arcs may form a complicated structure with serial and parallel dimensions. Referring now to FIGURE 2b, a flowchart of the same program illustrated in 2a is shown in which the program is expressed in terms of tasks and arcs. It may be observed that two sets of steps may be executed in parallel.

The DFL operating system allows processing jobs to be expressed in separate program modules for accomplishing separate tasks and for the necessary data flows between these tasks to be expressed as arcs. The DFL operating system then provides a mechanism for the required flows of data along arcs between the program modules whether or

not these modules happen to be on the same data processing unit. Additionally, the DFL operating system operates in accordance with a protocol for communication of data in separate pieces or "packets" as required for efficient communication between program modules.

The DFL operating system comprises a set of communication rules, a communication protocol, a management program and a set of system "calls" for communication between the program modules and associated routines for execution in response to the system calls. The communication rules comprise two lookup tables containing static reference information. The first lookup table comprises information about the tasks including a listing of the tasks by name and number, their priorities, the starting address of their code in memory and their stack size. The second lookup table comprises information about the arcs including a listing of the arcs by number, their source and destination, their size and whether they are internal or external to the processor unit to which they are assigned..

The communications protocol is a uniform structure for the data packets. In accordance with this structure the first four bytes in each data packet indicate its source task (application modules), the next four indicate its destination task, the next two indicate the length of the packet and the bytes following thereafter contain the actual data.

The management program is a group of instructions which keeps track of the status of each task (application module) on each processor unit and directs the processor to execute the appropriate tasks at appropriate times. The tasks are listed as either ready to run, blocked or

running. Tasks which are blocked need further data. Tasks which are ready are simply waiting for the required computer time to be executed. Tasks which are running are in the process of being executed. The management program lists the tasks in the ready category timewise by when they become ready and directs the processor to execute them in that order. Whenever specific operating system routines are completed the processor shifts to execute the management program which directs it to execute the next appropriate program.

The system calls comprise a language for interfacing between the program modules and the operating system. The system calls and their functions are shown in Table I.

TABLE I

Send (arcp, ptr, n) "arcp" = arc pointer - (name) "ptr" = memory address of packet n = size of packet	directs execution of the operating system output routine for data output to another task along the arc "arcp" for up to n elements of data starting at memory address "ptr"
Receive (arcp, ptr, maxn) "arcp" - arcpointer - (name) "ptr" - memory address allocated for packet n = size of packet	directs the execution of the operating system input routine for data input from another task along the arc "arcp" for up to n elements of data starting at memory address "ptr"

The computing system 10 also includes a DFL compiler program for "configuring" the DFL operating system by providing a medium for the user to supply the applications programs and by formulating the lookup tables of the communications rules in response to input form the user. The DFL compiler is resident within the RAM memory for the host computer 12 where it can be conveniently called up by the user when the system is being initially set up. The user furnishes the information for the lookup tables in the form of a set of standardized instructions defining the program name and its input and output characteristics (input/output data types and the program's "connectivity") which are included at the start of each application program module. The formats for these instructions are illustrated in Table II below for the programs "genl" and "Copy".

TABLE II

Gen 1 (in, out)	Task name
Input (char, in)	Input type
Output (char, out)	Output type
Copy (cin, cout)	Task name
Input (char, cin)	Input type
Output (char, cin)	Output type
Gen 1.cout ----- copy.cin	Arc
Copy.cout ------ gen 1.in	Arc

The lookup tables and required applications programs are downloaded to the processing units as a part of and along with the operation system.

Referring now to FIGURES 3 and 4 simplified flowcharts for the operating system routines which the processor is directed to execute in response to the Send and Receive system calls are shown. In particular, the flow charts are intended to illustrate the interaction between the operating system routines and various hardware components to be described later which are a part of the means for implementing data transfers over the system bus. FIGURE 3 shows the operating system Send routine. If the operating system determines that an interprocessor transfer is necessary in step 150 then a series are steps 152, 154, and 156 are undertaken whereby various chips are selected and initialization signals containing information required for data transfer operations are sent to those chips. When these signals are supplied, data transfer operations are initiated over the system bus. FIGURE 4 shows the operating system Receive routine. If the operating system determines in steps 160 and 162 that an interprocessor transfer is underway then a series of steps 164 and 166 are undertaken whereby various chips are selected and initialization signals containing information required for data transfer operations are sent to those chips. However, these signals only prepare a processor unit to receive a data transfer when contacted by a processor unit with data ready to be transmitted.

Referring now to FIGURE 6 a simplified block diagram of the hardware system architecture for data transfers is illustrated. The processor units 50, 52 and 54 and the I/O units 56 and 58 are interconnected by a system bus 60. The processor 50 is shown in somewhat

greater detail as a representative processor unit and includes bus address assignmemt logic 62, bus arbitration logic 64, data transmission logic 66, data buffer 68, microprocessor 70, RAM memory 72 and local bus 74. These components are implemented in hardware by via conventional registers, computers, buffers, logic circuits and other medium and small scale intergrated devices. The I/O unit 56 is also shown in greater detail as a representative unit including bus address assignment logic 76 and I/O interface logic 78 (although we are primarily concerned with transfers between processing units).

The bus address assignment logic 62 (and assignment logic 76) operates to determine a unique address ior each of the processor units (and I/O units). The assignment logic on each processor unit is connected in series with the assignment logic components for the processors on each adjacent side of the unit. When the power is first applied the address is determined by receiving and passing along a signal level on a special service line on the system bus in a daisy-chain manner.

The data transfer arbitration logic 64 functions to acquire access to and mastery of the system bus 60 on behalf of the data transmission logic 66 and the processor unit 50. The arbitration logic 64 is connected in series with the arbitration logic components for the processors on each adjacent side of the unit over special service lines to form a ring token network. A signal level in rapidly and continuously passed around the network from one processor unit to another. However, when the signal level reaches a processor unit which requires access to the system bus the signal level is retained and exclusive control over the system bus is assumed by the data transmission logic 66.

The data buffer 68 functions to enable a communications pathway from the system bus 60 to the local bus 74 in response to signals from the data transmission logic 66. The RAM memory 72 stores data and programs for use and execution in conventional fashion by the microprocessor 70. The data transmission logic 66 functions to implement and control data transfers between processors in cooperation with the applications and operating system programs stored in the RAM memory 72 as will be further explained hereinafter.

Application program modules are executed by the microprocessor 70 in the conventional manner except at point when data is required as input or data is ready for output. At such points the applications programs include system call instructions which direct the microprocessor 70 to execute operating system routines for data input and output between application modules (tasks) in accordance with the connectivity between these modules (the arcs) specified in the operating system. The steps in the operating system routines are as illustrated in FIGURES 3 and 4 and previously described. The routine executed in response to a Send call when a interprocessor transfer is indicated supplies the initialization signals for the data transmission logic. These signals represent chip selection instructions, the starting address for the actual data in the RAM memory 72, the destination address for the processor to which the data is to be sent, the number of elements of data to be sent and a request to send data. These signals are enabled onto the local bus 74 where they may be latched into the appropriate chips and registers within the data

transmission logic in accordance with the chip select instructions.

Referring now to FIGURES 5 and 6, the sender logic and the receiver logic are illustrated in greater detail showing the basic functional components within the data transmission logic 66 and the connections between functional components which may represent one or more actual chips and that certain chips may incorporate more than one function.

The sender logic includes a chip select logic 100 which in response to instruction signals from the operating system generates signals to selectively activate other chips containing memory functions within the system. In particular the chip select logic 100 actuates the send flag register 102, the data address register and counter 104, the count register and send done logic 106, and the destination address register 108. Upon activation by the chip select logic 100 the data address register and counter 104 receives the more significant part (eight bits) of starting address or page address in the RAM memory of the sending processor oi the data to be transferred. During actual transfers the counter increments this address in coordination with signals from the sequence controller 108 in order to generate a full address including the less significant part of the address. The resulting sequential addresses are provided to the address buffer 110 where they are enabled onto the local bus 74a in coordination with signals from the sequence controller 108 and are provided over the system bus 60 to an address buffer for the receiver logic on the destination processor. The receiver logic on the destination processor

responds on a different special line on the system bus 60 with an acknowledgment signal if it is available and prepared to receive a data packet. When the poll logic 116 receives this acknowledgment signal it in turn provides a signal to the sequence controller 108 and the local bus control logic 118. In response to these signals the local bus control logic 118 interrupts the operations of the processor so that the data transfer can take place over the local bus 74a, and the sequence controller 108 provides control and timing signals to the RAM memory for the sending processor, to the counter in the data address register and counter 104, to the address buffer 110, to the data buffer 120 and the over a special service line on the system bus 60 to the sequence controller for the receiver logic. The sequence controller 108 provides the required signals to coordinate the operations of the sender logic during data transfers so that the RAM memory addresses are enabled onto the local bus 74 and the data enabled onto the system bus 60 in proper sequence.

The receiver logic includes many components having parallel or complimentary functions with similar components of the sender logic and which are frequently implemented on the same chips. In particular, the receiver logic includes chip select logic 130 (on the same chip as logic 100) which in response to instruction signals from the operating system genrates signals to selectively activate other chips containing memory functions. The chip select logic 130 activates the receiver available register 132, and the data address register 134. Upon activation by the chip select logic 130 the data address register 134 receives the more significant part of the starting address or page address

in RAM memory for the receiving processor of the memory locations allocated for the data to be received. This memory address is then supplied to the address buffer 136 during data transfer operations where it is combined with a series of progressively incremented partial addresses (less significant parts) provided over the system bus 60 from the counter (of the data address register and counter 104) of the sender logic for the sending processor. When combined together the partial address information from the data address register 134 and the sending processor provide complete data memory addresses which can be enabled onto the local bus by the address buffer 136 for use by the receiving processor's RAM memory. Upon activation by the chip select logic 130 the receiver available register 132 receives an available flag signal from the operating system which sets an available flag in the register. The available flag indicates by its signal level to the poll logic 138 that the processor associated with the transmission logic is available to receive a data transfer. The register and destination comparator 140 receives destination address signals from other processors over special control lines reserved for this purpose on the system bus 60. These signals are compared to the address of the processor with which the receiving logic is associated as provided by the address assignment logic 146 and stored in a register for the device 140. When these addresses match the register and destination comparator 140 supplies a signal to the local bus control logic 142. The local bus control logic 142 then interrupts the processor's operations so that a data transfer can take place and provides a signal level indicative of this interruption to the poll logic 138.

When the poll logic 138 is in receipt of signals from both the receiver available register 132 and the local bus control logic 142 it acknowledges the receiving processors availability to receive data to the sending processor by placing a signal level on a special service line on the system bus which connects with the poll logic on the sending processor. The sequence controller 144 is then actuated by signals which are received over other service liens on the system bus from the sequence controller on the sending processor. The sequence controller 144 supplies the signals to coordinate to operations of the receiver logic during data transfers so that the RAM memory addresses are enabled onto the local bus 74b and the data enabled onto the system bus 60 by the data buffer 148 in proper sequence. When the data transfer is complete and the sending processor releases the system bus the receiver done logic 152 detects the change in signal levels in the system bus arbitration logic 150. The receiver done logic 152 then provides a signal to the receiving processor for it to resume normal operations.

WHAT IS CLAIMED IS:

1. A system for effectuating data communications in a multiprocessor computer system having a common bus connecting the processors, comprising a plurality of individual processor units including:

microprocessor means for executing program instructions;

memory means for storing data and instructions to be processed and executed by the microprocessor means;

means for securing exclusive access to the common bus by interacting with a bus arbitration system;

buffer means for enabling a communications pathway between the processor unit and the common bus;

means for signaling to and form selected processors for establishing and acknowledging availability for data communications;

means separate from said microprocessor means for supplying data memory address signals to the memory means and onto the common bus;

means for providing signals adapted for initializing and initiating a data transfer sequence including transfer request, data memory address and processor destination address signals;

sequence controller means for supplying timing and control signals for tarnsferring data over the common bus from one processor unit to another.

2. The system of claim 1 wherein said means for providing signals for initializing and initiating a data transfer, comprises:

operating system program means resident within the memory means for providing the signals upon execution by the microprocessor means in response to system call instructions included in application data processing program modules.

3. The system of claim 1 wherein said means for signalling, comprises:

register means connected to the common bus for supplying and receiving destination address signals;

comparator means for comparing destination address sginals with processor address information;

logic means for providing signals to establish and acknowledge data communciations availability.

4. A multiprocessing computer system, comprising: a plurality of processor units having:

(a) microprocessor means for executing program instructions,

(b) memory means for storing data and program including applications program modules for processing data and an operating system program routines for helping to effectuate the communication of data between different applications program modules which may be resident on different processor units;

(c) a local bus means connecting said microprocessor means and said memory means;

a system bus for interconnecting said plurality of processor units;

means for arbitrating access to said system bus between said processor units;

a plurality of buffer means, one of which is associated with each processor unit, for connecting the local bus associated with each such processor unit to the system bus and enabling a communications pathway from its processor unit onto said system bus;

a plurality of data transmission logic means, one of which is associated with each processor unit and which are connected to the system bus and the local bus and the buffer means for the processor unit with which each logic means is associated, each such logic means operative for implementing data communications between processor units over said system bus in response to signals generated by the execution of said operating system routines which are undertaken as a result of the execution of system call instructions included in said applications program modules.

5. The system of claim 4 wherein the data stored in said memory means and transferred over said system bus is stored and transferred in packets.

6. The system of claim 5 wherein said data transmission logic means comprises:

means for signaling to and from selected processor units on special address and control lines for establishing and acknowledging availability for data communications;

means separate from said microprocessor means for supplying data memory address signals to the memory means of the processor unit with which the logic means is associated and onto the system bus;

sequence controller means for supplying timing and control signals for transferring data over the system bus from one processor unit to another.

7. The system of claim 6, wherein said means for supplying data memory address signals comprises a register which receives the more significant part for the memory address signals from the processor unit with which it is associated and a counter for generating the less significant part for each memory address in accordance with signals from the sequence controller means.

8. A method for effectuating data transfer between processor units in a multiprocessor system having a common bus connecting the processor units, comprising the steps of:

programming a data processing job into separate tasks which are adapted for executing discrete parts of the processing job on different processor units and arc information which defines the connectivity for data flow purposes between the tasks;

implementing data transfers along the arcs between tasks on different processor units by transferring data in packets over the common bus in accordance with the following steps for each transfer of data from a source processor unit to a destination processor unit:

(a) referencing the arc information at the source processor unit for a specific data transfer,

(b) setting a register on the source processor unit as a flag to indicate that a data packet is ready for transfer over the common bus,

(c) securing access to the common bus for the source processor unit by interacting with a menas for arbitrating access to the common bus in response to the register flag,

(d) supplying a processor destination address from the source processor unit in accordance with the arc information onto the common bus for establishing contact between the source processor unit and the destination processor unit,

(e) signalling between the source processor unit and the destination processor unit to acknowledge availability for data transfers;

(f) enabling a communications pathway from the source processor unit over the common bus to the destination processor unit;

(g) supplying data memory address, control and timing signals for reading data from the memory means associated with the source processor unit onto the common bus and writing data on the common bus into the memory associated with the destination processor unit.

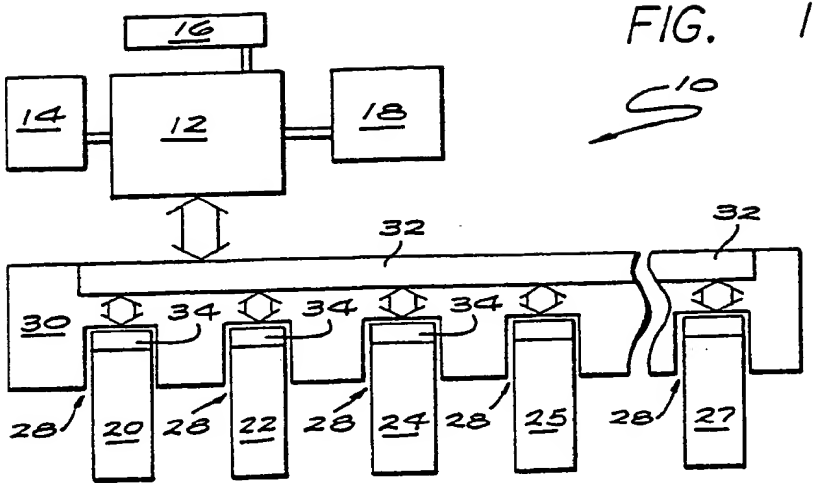

16
14
12
18
10
32
32
30
34
34
34
28
20
28
22
28
24
28
25
28
27

FIG. 1

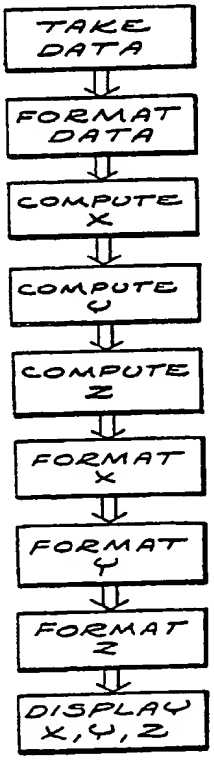
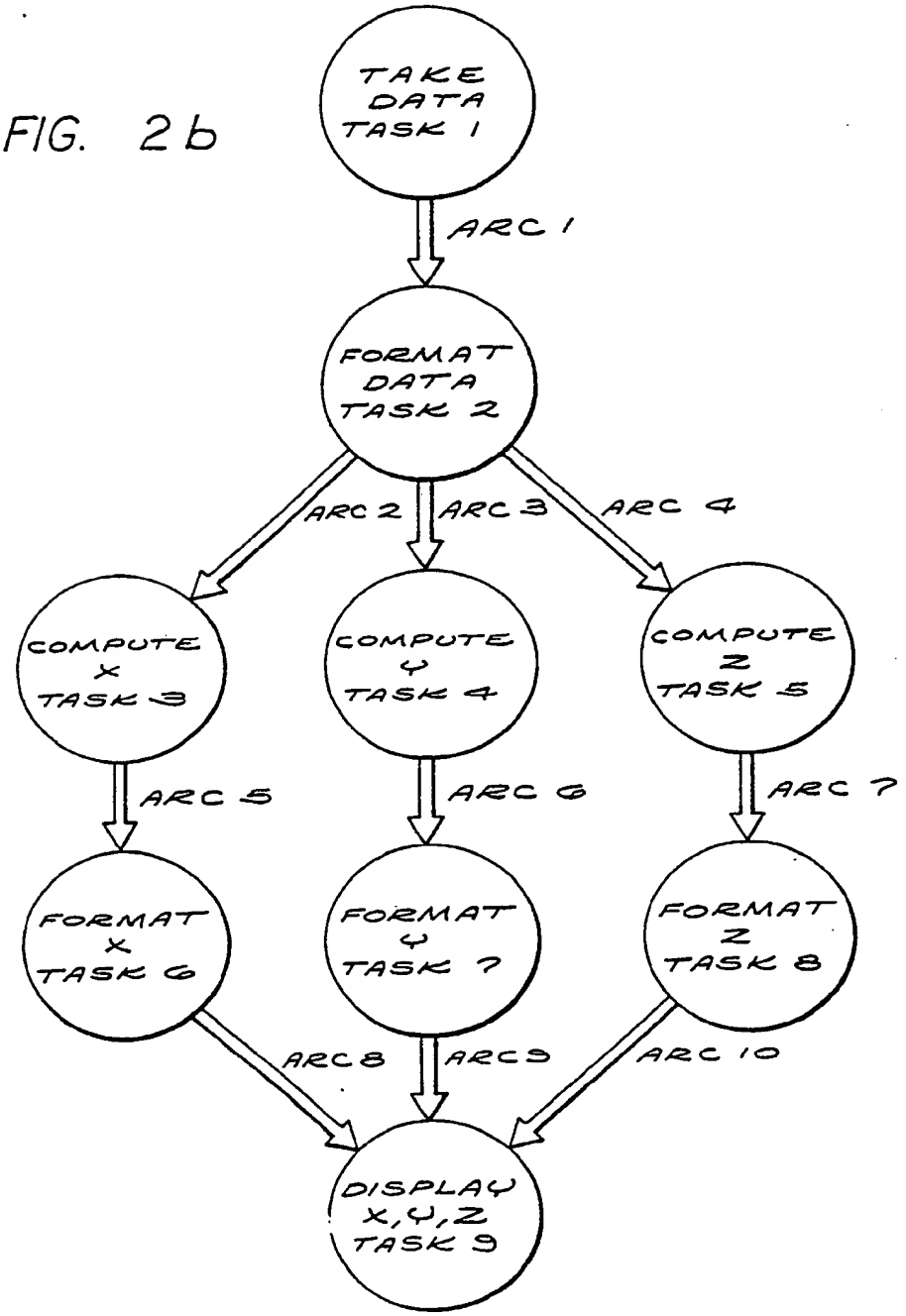

TAKE DATA
FORMAT DATA
COMPUTE X
COMPUTE Y
COMPUTE Z
FORMAT X
FORMAT Y
FORMAT Z
DISPLAY X,Y,Z

FIG. 2a

FIG. 2b


TAKE DATA TASK 1
ARC 1
FORMAT DATA TASK 2
ARC 2
ARC 3
ARC 4
COMPUTE X TASK 3
COMPUTE Y TASK 4
COMPUTE Z TASK 5
ARC 5
ARC 6
ARC 7
FORMAT X TASK 6
FORMAT Y TASK 7
FORMAT Z TASK 8
ARC 8
ARC 9
ARC 10
DISPLAY X,Y,Z TASK 9

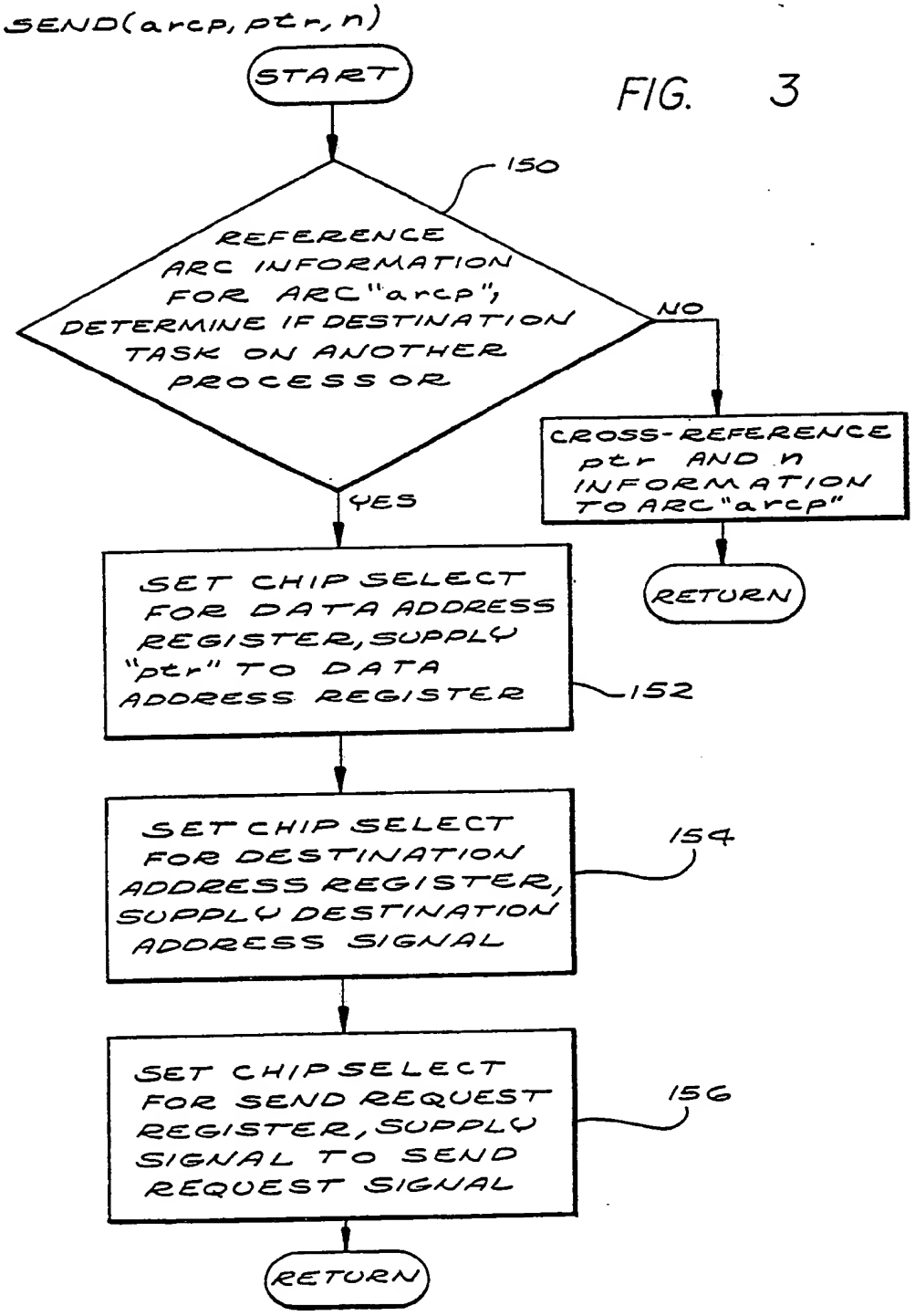

SEND(arcp, ptr, n)
START
150
REFERENCE ARC INFORMATION FOR ARC "arcp", DETERMINE IF DESTINATION TASK ON ANOTHER PROCESSOR
NO
YES
CROSS-REFERENCE ptr AND n INFORMATION TO ARC "arcp"
RETURN
SET CHIP SELECT FOR DATA ADDRESS REGISTER, SUPPLY "ptr" TO DATA ADDRESS REGISTER
152
SET CHIP SELECT FOR DESTINATION ADDRESS REGISTER, SUPPLY DESTINATION ADDRESS SIGNAL
154
SET CHIP SELECT FOR SEND REQUEST REGISTER, SUPPLY SIGNAL TO SEND REQUEST SIGNAL
156
RETURN

FIG. 3

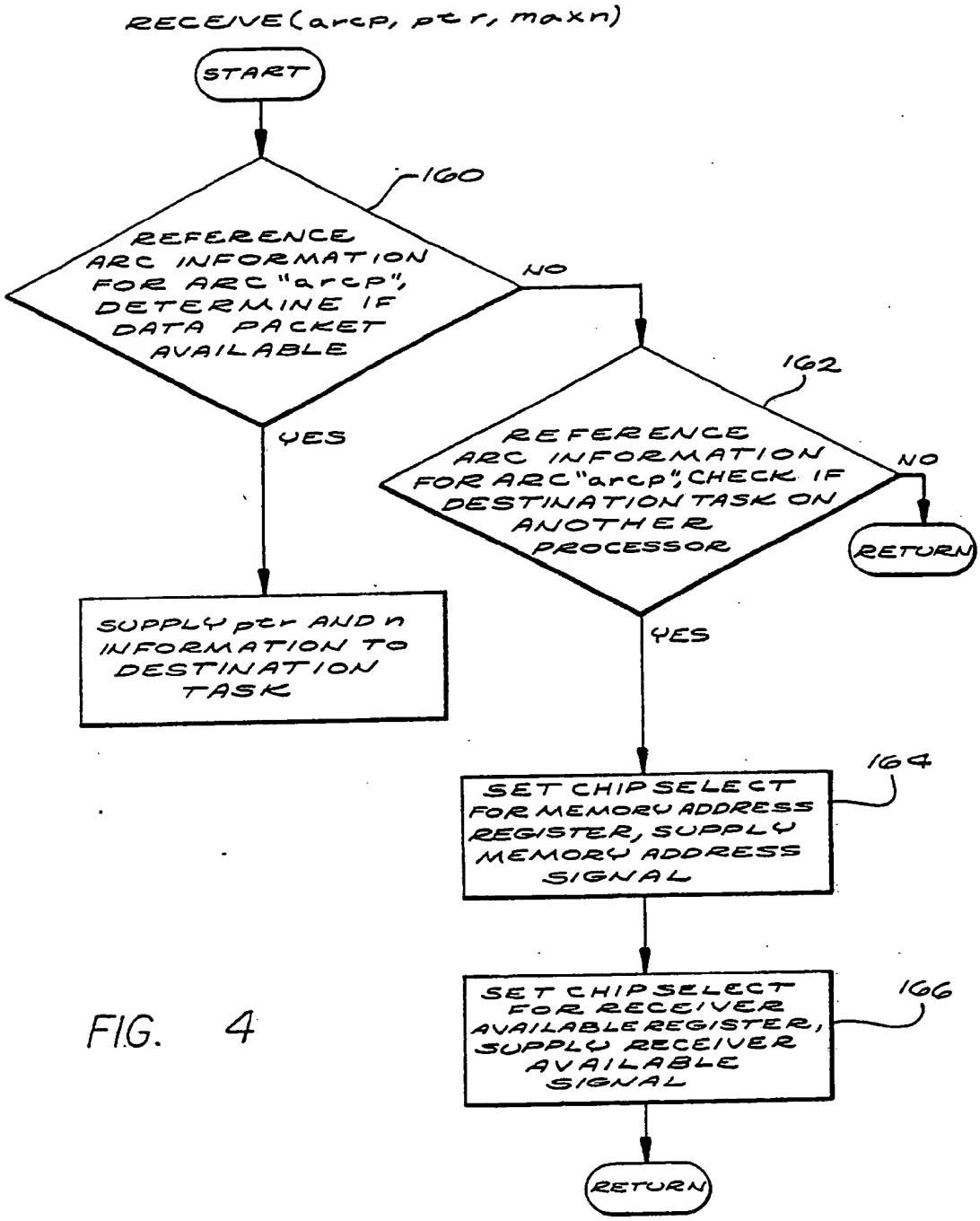

RECEIVE(arcp, ptr, maxn)
START
160
REFERENCE ARC INFORMATION FOR ARC "arcp", DETERMINE IF DATA PACKET AVAILABLE
NO
YES
162
REFERENCE ARC INFORMATION FOR ARC "arcp", CHECK IF DESTINATION TASK ON ANOTHER PROCESSOR
NO
RETURN
YES
SUPPLY ptr AND n INFORMATION TO DESTINATION TASK
164
SET CHIP SELECT FOR MEMORY ADDRESS REGISTER, SUPPLY MEMORY ADDRESS SIGNAL
166
SET CHIP SELECT FOR RECEIVER AVAILABLE REGISTER, SUPPLY RECEIVER AVAILABLE SIGNAL
RETURN

FIG. 4

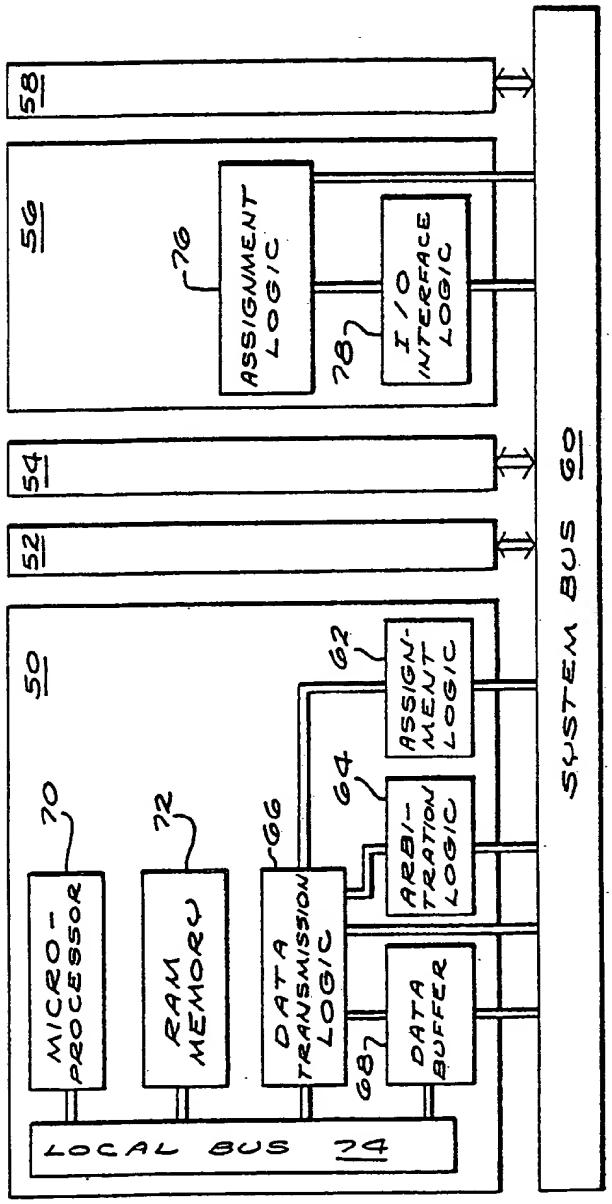
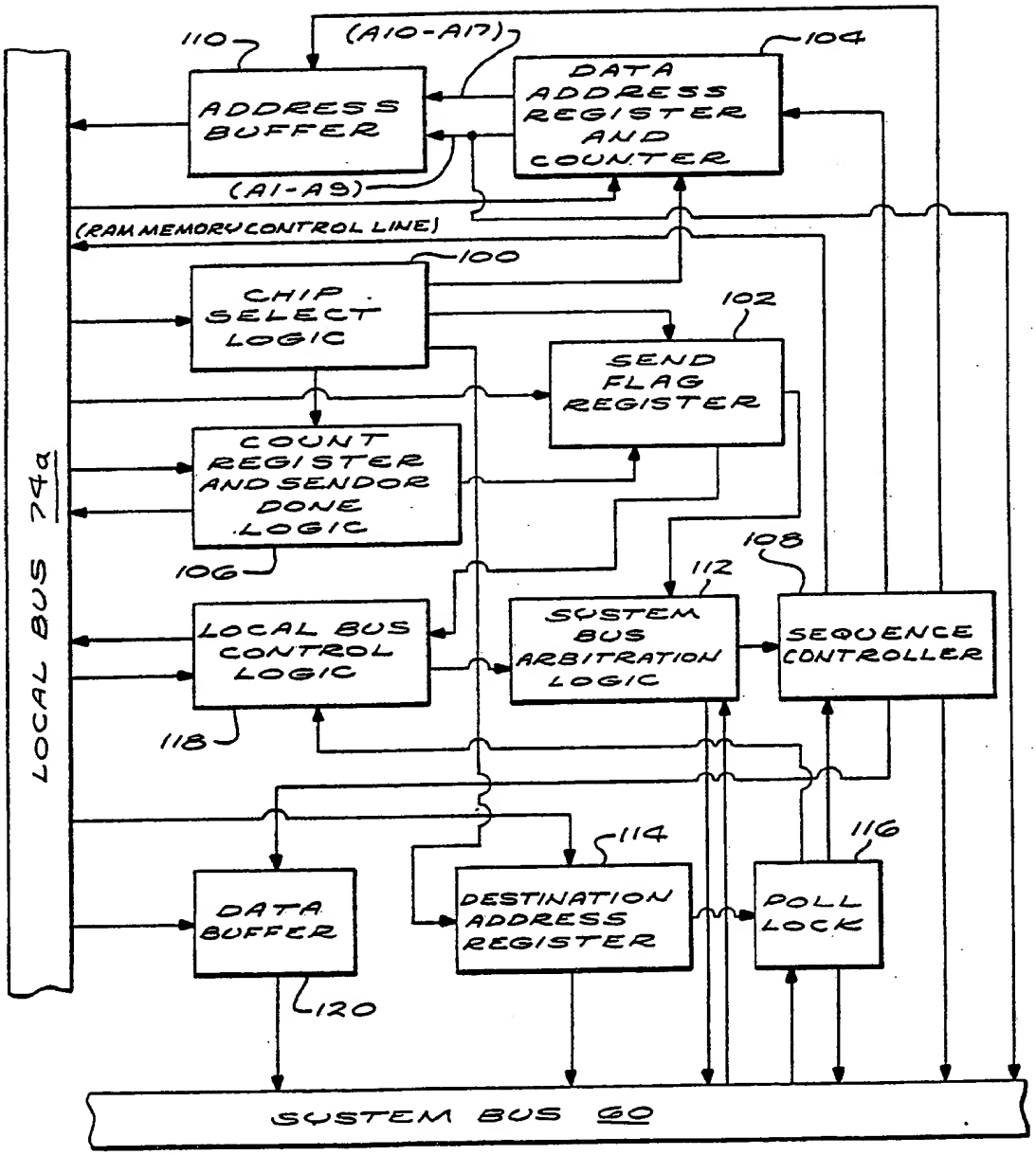

58
56
76
ASSIGNMENT LOGIC
78
I/O INTERFACE LOGIC
54
52
50
62
ASSIGN-MENT LOGIC
64
ARBI-TRATION LOGIC
70
MICRO-PROCESSOR
72
RAM MEMORY
66
DATA TRANSMISSION LOGIC
68
DATA BUFFER
LOCAL BUS 74
SYSTEM BUS 60

FIG. 5



SENDER LOGIC

FIG. 6

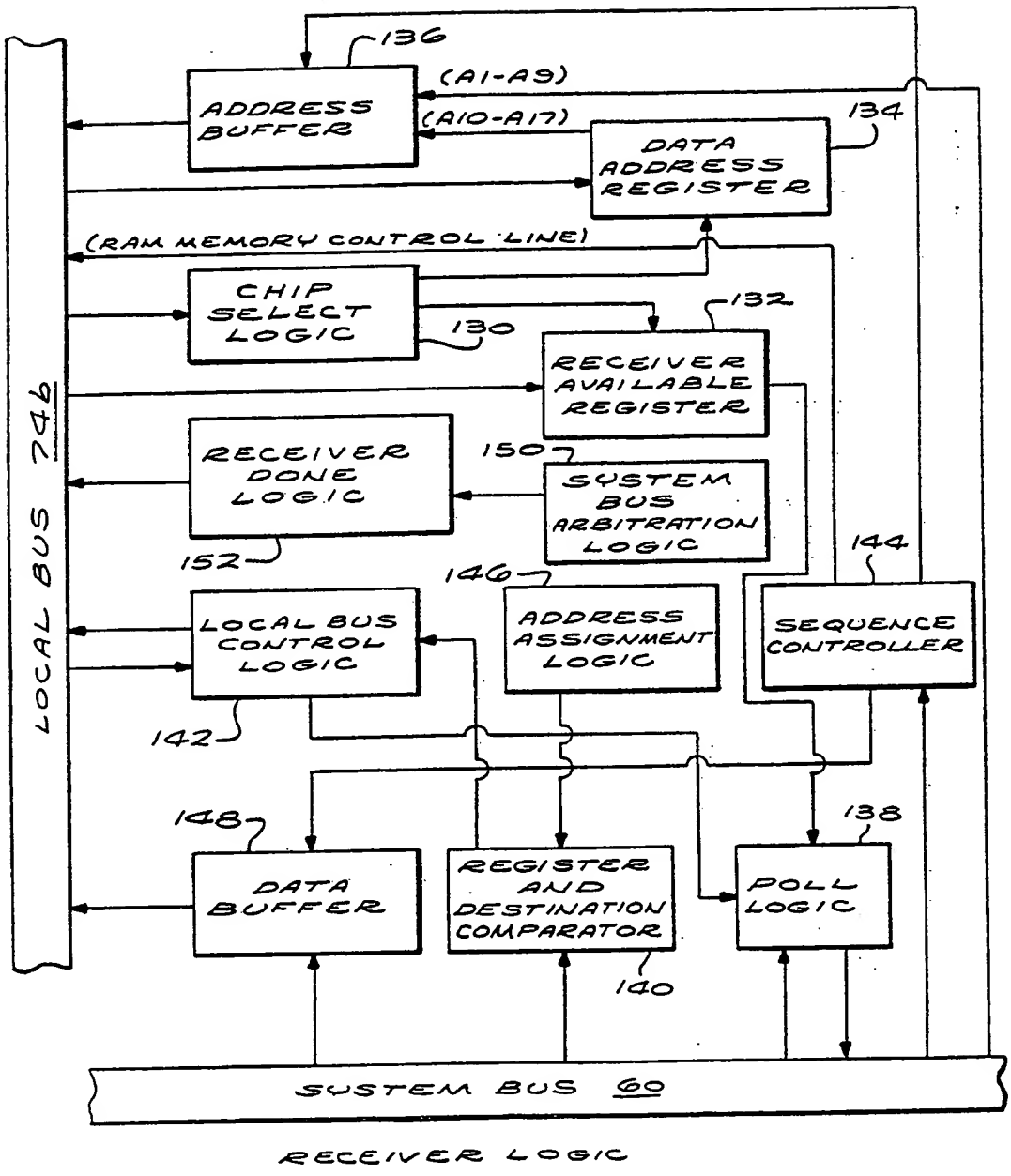

136
(A1-A9)
ADDRESS BUFFER
(A10-A17)
134
DATA ADDRESS REGISTER
(RAM MEMORY CONTROL LINE)
CHIP SELECT LOGIC
130
132
RECEIVER AVAILABLE REGISTER
LOCAL BUS 746
RECEIVER DONE LOGIC
150
SYSTEM BUS ARBITRATION LOGIC
152
144
146
LOCAL BUS CONTROL LOGIC
ADDRESS ASSIGNMENT LOGIC
SEQUENCE CONTROLLER
142
148
138
DATA BUFFER
REGISTER AND DESTINATION COMPARATOR
POLL LOGIC
140
SYSTEM BUS 60
RECEIVER LOGIC

FIG. 7

INTERNATIONAL SEARCH REPORT

International Application No. PCT/US88/01236

I. CLASSIFICATION OF SUBJECT MATTER (if several classification symbols apply, indicate all) [6]

According to International Patent Classification (IPC) or to both National Classification and IPC

IPC(4): G06F 13/36
U.S. Cl. 364/200

II. FIELDS SEARCHED

Minimum Documentation Searched [7]

Classification System	Classification Symbols
U.S.	364/200, 900

Documentation Searched other than Minimum Documentation to the Extent that such Documents are Included in the Fields Searched [8]

III. DOCUMENTS CONSIDERED TO BE RELEVANT [9]

Category *	Citation of Document, [11] with indication, where appropriate, of the relevant passages [12]	Relevant to Claim No. [13]
X	US, A, 4,320,467 (GLASS) 16 March 1982, see entire document.	1-8
X	US, A, 4,601,586 (BAHR ET AL.) 22 July 1986, see entire document.	1-8
E, X	US, A, 4,744,023 (WELSCH) 10 May 1988, see entire document.	1-8
Y	US, A, 4,363,093 (DAVIS ET AL.) 07 December 1982, see columns 43-48.	1-8
Y	US, A, 4,363,094 (KAUL ET AL.) 07 December 1982, see columns 3-5.	1-8
Y	US, A, 4,404,628 (ANGELO) 13 September 1983, see entire document.	1-8
Y	US, A, 4,470,114 (GERHOLD) 04 September 1984, see entire document.	1-8

* Special categories of cited documents: [10]

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

IV. CERTIFICATION

Date of the Actual Completion of the International Search	Date of Mailing of this International Search Report
28 JUNE 1988	10 AUG 1988
International Searching Authority	**Signature of Authorized Officer**
ISA/US	Debra Chun DEBRA CHUN

FURTHER INFORMATION CONTINUED FROM THE SECOND SHEET

A	US, A, 3,735,365 (NAKAMURA ET AL.) 22 May 1973, see columns 2-4.	1-8
A	US, A, 4,375,639 (JOHNSON, JR.) 01 March 1983, see entire document.	1-8
A	US, A, 4,387,425 (EL-GOHARY) 07 June 1983, see entire document.	1-8
A	US, A, 4,543,626 (BEAN ET AL.) 24 September 1985, see entire document.	8

V. ☐ OBSERVATIONS WHERE CERTAIN CLAIMS WERE FOUND UNSEARCHABLE [1]

This international search report has not been established in respect of certain claims under Article 17(2) (a) for the following reasons:

1. ☐ Claim numbers ; because they relate to subject matter [1] not required to be searched by this Authority, namely:

2. ☐ Claim numbers , because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out [1], specifically:

3. ☐ Claim numbers_____, because they are dependent claims not drafted in accordance with the second and third sentences of PCT Rule 6.4(a).

VI. ☐ OBSERVATIONS WHERE UNITY OF INVENTION IS LACKING [2]

This International Searching Authority found multiple inventions in this international application as follows:

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims of the international application.

2. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims of the international application for which fees were paid, specifically claims:

3. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claim numbers:

4. ☐ As all searchable claims could be searched without effort justifying an additional fee, the International Searching Authority did not invite payment of any additional fee.

Remark on Protest

☐ The additional search fees were accompanied by applicant's protest.

☐ No protest accompanied the payment of additional search fees.

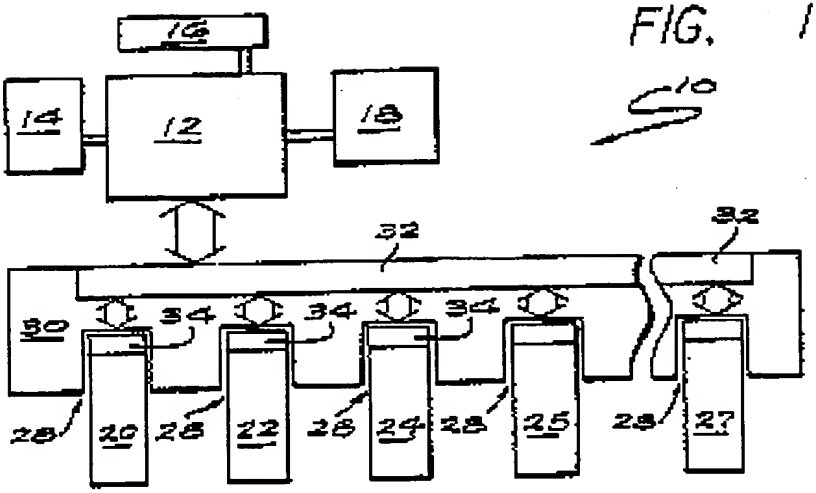

16
14
12
18
10
32
32
30
34
34
34
28
20
28
22
28
24
28
25
28
27

FIG. 1

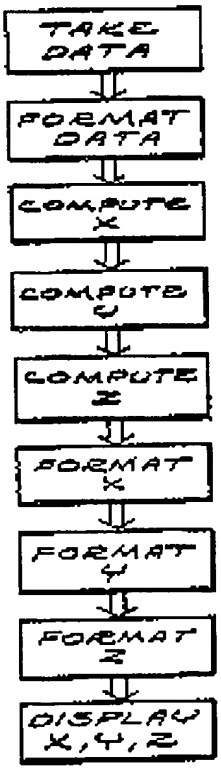
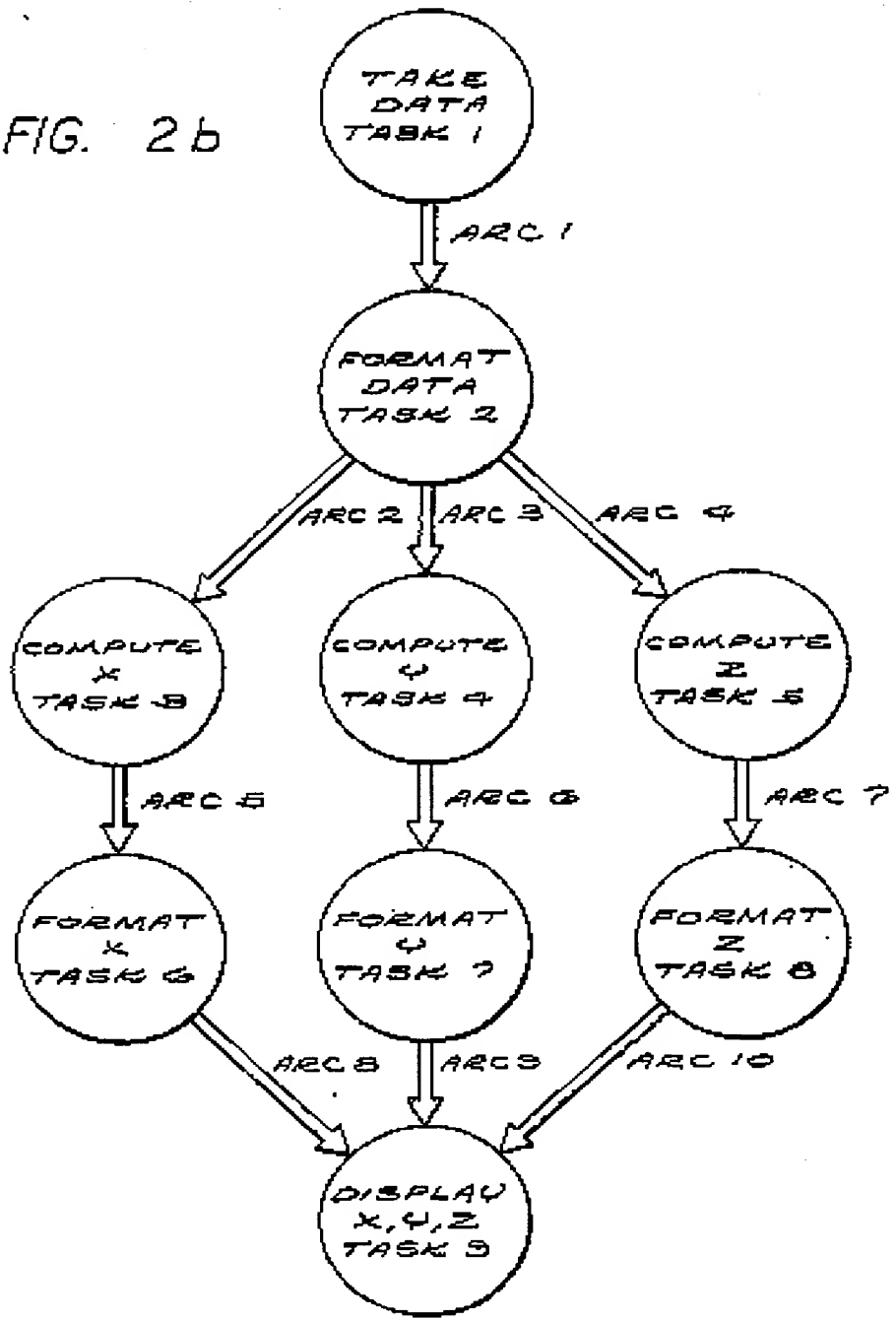

TAKE DATA
FORMAT DATA
COMPUTE X
COMPUTE Y
COMPUTE Z
FORMAT X
FORMAT Y
FORMAT Z
DISPLAY X, Y, Z

FIG. 2a

FIG. 2b


TAKE DATA TASK 1
ARC 1
FORMAT DATA TASK 2
ARC 2
ARC 3
ARC 4
COMPUTE X TASK 3
COMPUTE Y TASK 4
COMPUTE Z TASK 5
ARC 5
ARC 6
ARC 7
FORMAT X TASK 6
FORMAT Y TASK 7
FORMAT Z TASK 8
ARC 8
ARC 9
ARC 10
DISPLAY X,Y,Z TASK 9

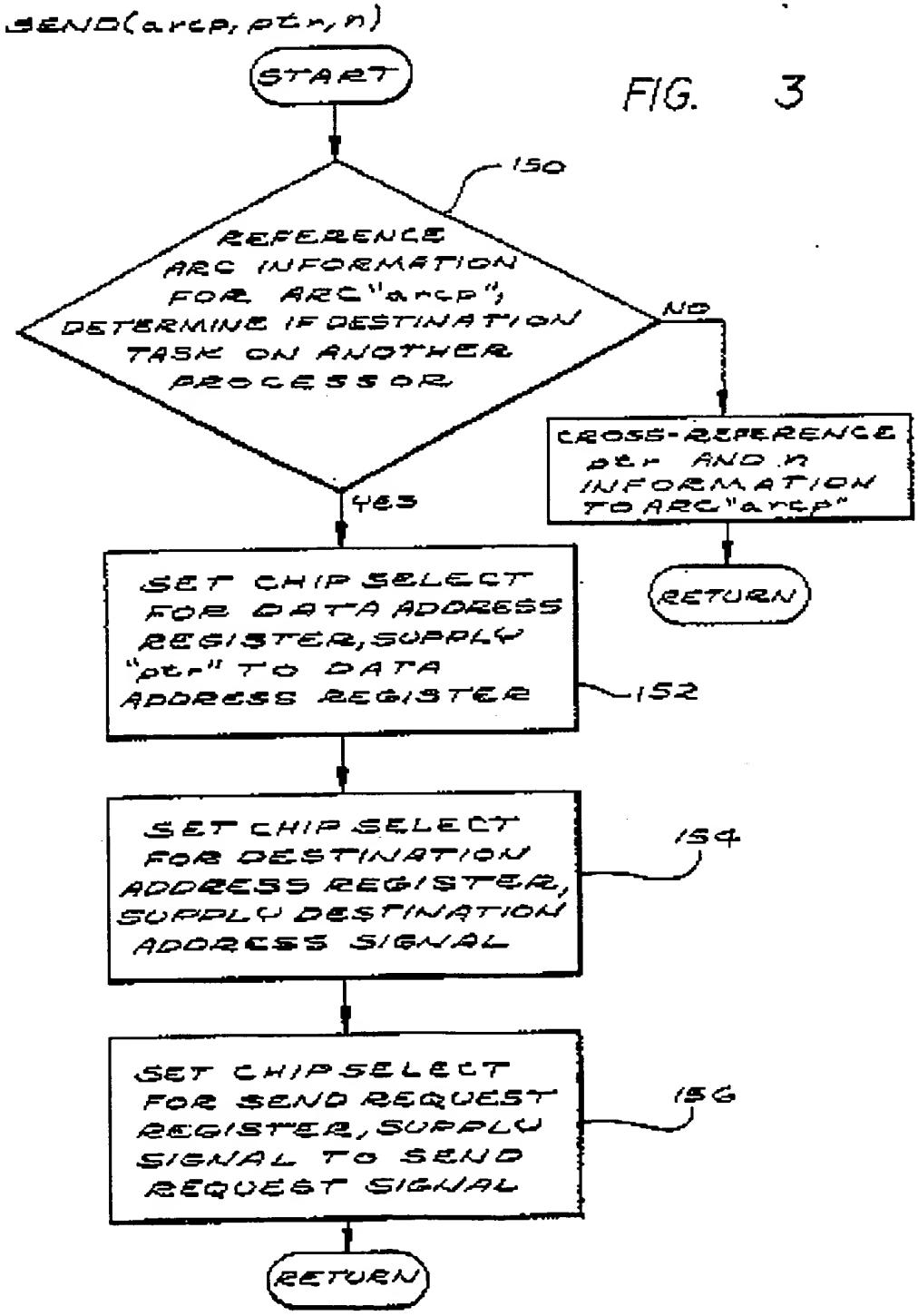

SEND(arcp, ptr, n)
START
150
REFERENCE ARC INFORMATION FOR ARC "arcp", DETERMINE IF DESTINATION TASK ON ANOTHER PROCESSOR
NO
CROSS-REFERENCE ptr AND n INFORMATION TO ARC "arcp"
RETURN
YES
SET CHIP SELECT FOR DATA ADDRESS REGISTER, SUPPLY "ptr" TO DATA ADDRESS REGISTER
152
SET CHIP SELECT FOR DESTINATION ADDRESS REGISTER, SUPPLY DESTINATION ADDRESS SIGNAL
154
SET CHIP SELECT FOR SEND REQUEST REGISTER, SUPPLY SIGNAL TO SEND REQUEST SIGNAL
156
RETURN

FIG. 3

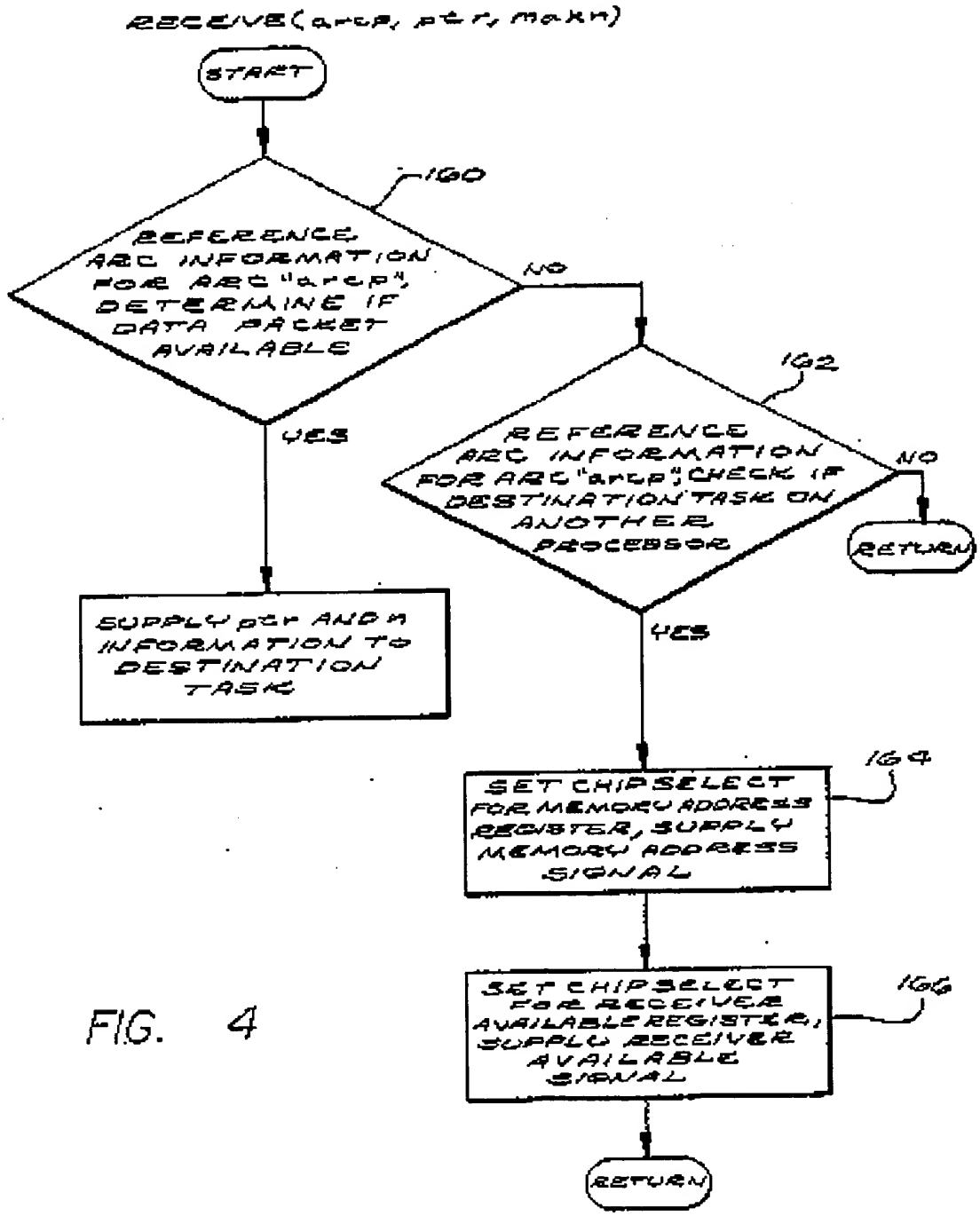

RECEIVE(arcp, ptr, maxn)
START
160
REFERENCE ARC INFORMATION FOR ARC "arcp", DETERMINE IF DATA PACKET AVAILABLE
NO
YES
162
REFERENCE ARC INFORMATION FOR ARC "arcp"; CHECK IF DESTINATION TASK ON ANOTHER PROCESSOR
NO
RETURN
YES
SUPPLY ptr AND n INFORMATION TO DESTINATION TASK
164
SET CHIPSELECT FOR MEMORY ADDRESS REGISTER, SUPPLY MEMORY ADDRESS SIGNAL
166
SET CHIPSELECT FOR RECEIVER AVAILABLE REGISTER, SUPPLY RECEIVER AVAILABLE SIGNAL
RETURN

FIG. 4

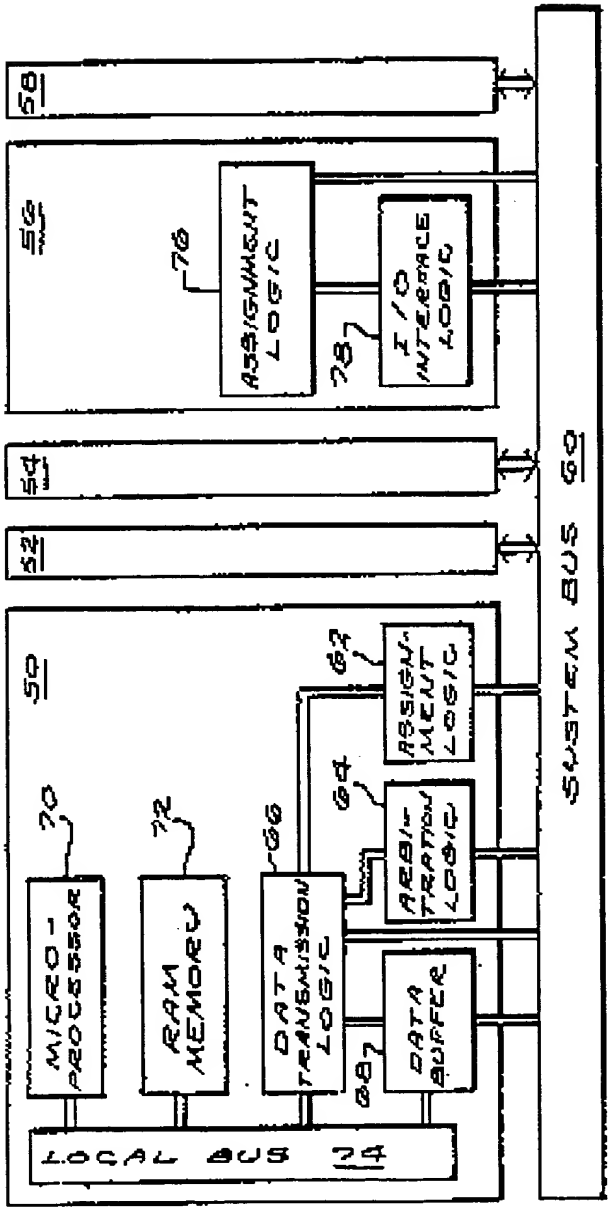
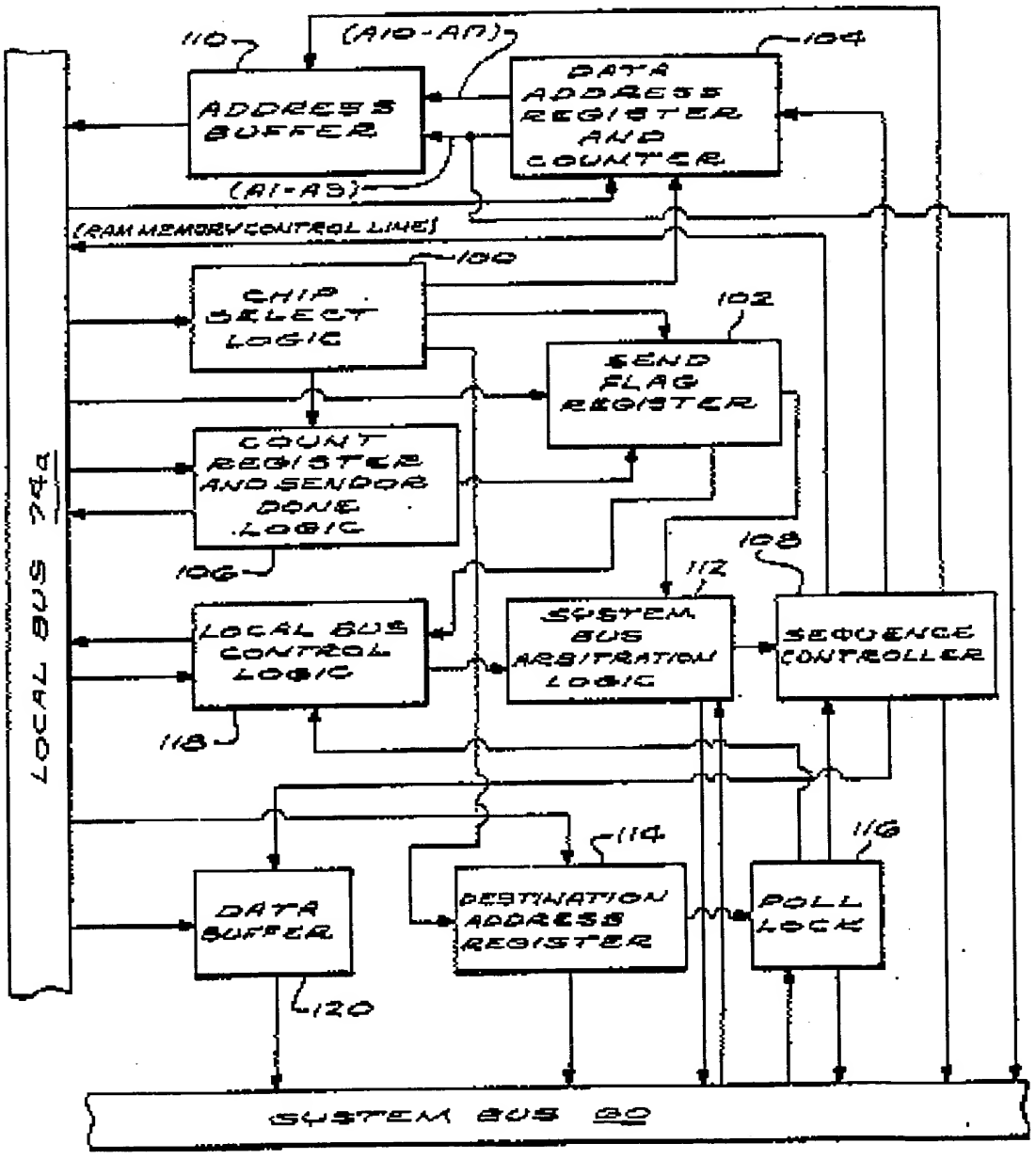

58
56
76
ASSIGNMENT LOGIC
78
I/O INTERFACE LOGIC
54
52
50
62
ASSIGN-MENT LOGIC
64
ARBI-TRATION LOGIC
66
DATA TRANSMISSION LOGIC
68
DATA BUFFER
70
MICRO-PROCESSOR
72
RAM MEMORY
LOCAL BUS 14
SYSTEM BUS 60

FIG. 5



SENDER LOGIC

FIG. 6

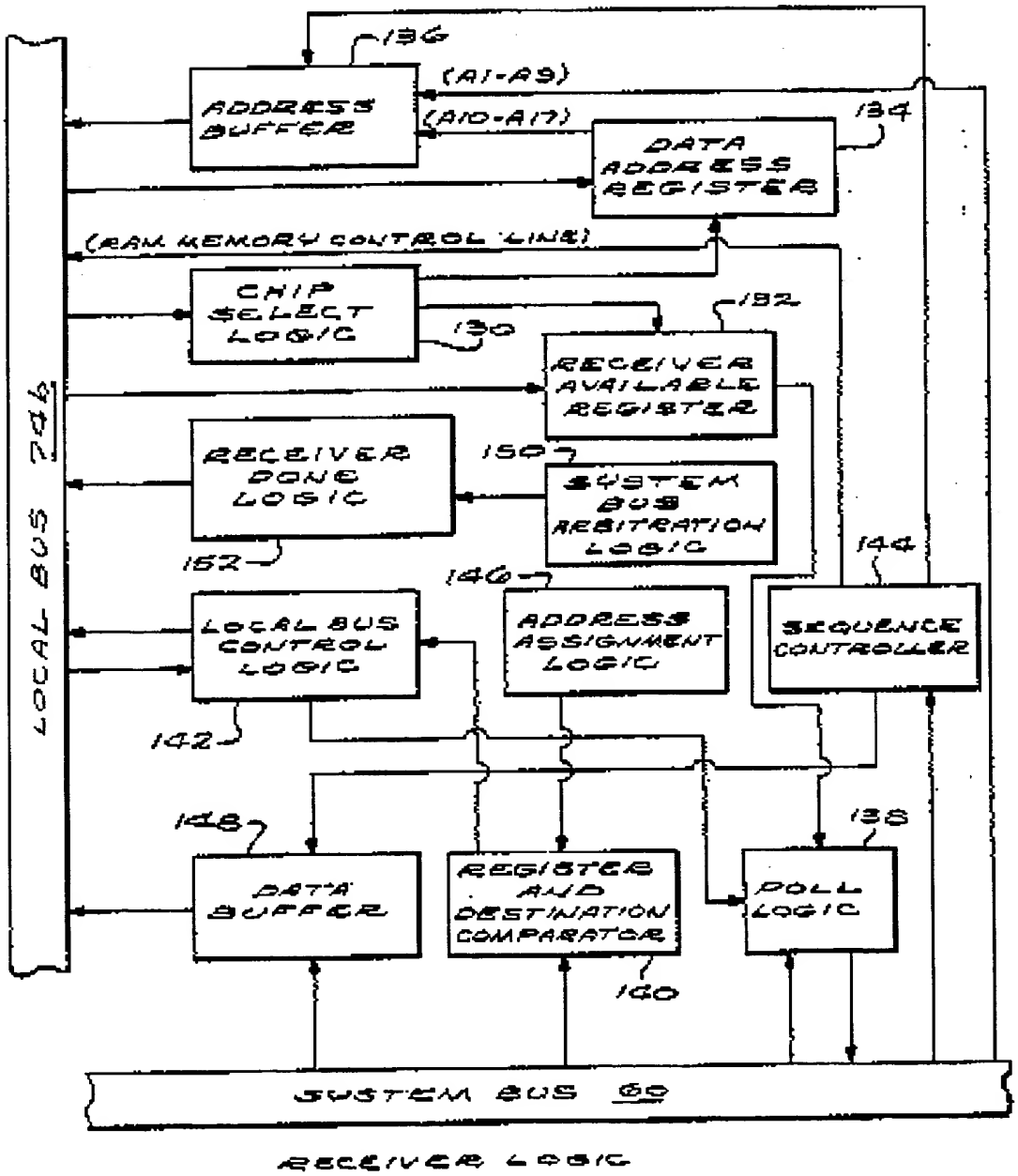

136
(A1-A9)
ADDRESS BUFFER
(A10-A17)
134
DATA ADDRESS REGISTER
(RAM MEMORY CONTROL LINE)
CHIP SELECT LOGIC
130
132
RECEIVER AVAILABLE REGISTER
LOCAL BUS 24
RECEIVER DONE LOGIC
150
SYSTEM BUS ARBITRATION LOGIC
152
144
140
LOCAL BUS CONTROL LOGIC
ADDRESS ASSIGNMENT LOGIC
SEQUENCE CONTROLLER
142
148
138
DATA BUFFER
REGISTER AND DESTINATION COMPARATOR
POLL LOGIC
146
SYSTEM BUS 60
RECEIVER LOGIC

FIG. 7